Q3/HW3 STAT 3701 R Package

Benjamin Nguyen

2018-03-02

This vignette is documented to demonstrate the innards of the my package "NguyenTools". Inside the package are datasets and functions. The following documentation will illustrate rudimentary ways to use the data sets and the functions.

```
library(ggplot2)
library(magrittr)
library(NguyenTools)
```

Demonstration of using functions in package

The following functions were provided in Homework 1 and Quiz 1 of STAT 3701.

The function 'continentAsia()' is a wrapper that uses the dataset 'gp2007', which will be described later down in the vignette. It filters the data by continents; namely it pulls out the part of the data in which the continent is 'Asia.'

```
continentAsia()
```

```
##      X            country continent lifeExp        pop  gdpPercap
## 1    1        Afghanistan      Asia  43.828   31889923   974.5803
## 2    8            Bahrain      Asia  75.635     708573 29796.0483
## 3    9         Bangladesh      Asia  64.062  150448339  1391.2538
## 4   19           Cambodia      Asia  59.723   14131858  1713.7787
## 5   25              China      Asia  72.961 1318683096  4959.1149
## 6   56   Hong Kong, China      Asia  82.208    6980412 39724.9787
## 7   59              India      Asia  64.698 1110396331  2452.2104
## 8   60          Indonesia      Asia  70.650  223547000  3540.6516
## 9   61               Iran      Asia  70.964   69453570 11605.7145
## 10  62               Iraq      Asia  59.545   27499638  4471.0619
## 11  64             Israel      Asia  80.745    6426679 25523.2771
## 12  67              Japan      Asia  82.603  127467972 31656.0681
## 13  68             Jordan      Asia  72.535    6053193  4519.4612
## 14  70   Korea, Dem. Rep.      Asia  67.297   23301725  1593.0655
## 15  71        Korea, Rep.      Asia  78.623   49044790 23348.1397
## 16  72             Kuwait      Asia  77.588    2505559 47306.9898
## 17  73            Lebanon      Asia  71.993    3921278 10461.0587
## 18  79           Malaysia      Asia  74.241   24821286 12451.6558
## 19  84           Mongolia      Asia  66.803    2874127  3095.7723
## 20  88            Myanmar      Asia  62.069   47761980   944.0000
## 21  90              Nepal      Asia  63.785   28901790  1091.3598
## 22  97               Oman      Asia  75.640    3204897 22316.1929
## 23  98           Pakistan      Asia  65.483  169270617  2605.9476
## 24 102        Philippines      Asia  71.688   91077287  3190.4810
## 25 110       Saudi Arabia      Asia  72.777   27601038 21654.8319
## 26 114          Singapore      Asia  79.972    4553009 47143.1796
## 27 120          Sri Lanka      Asia  72.396   20378239  3970.0954
## 28 125              Syria      Asia  74.143   19314747  4184.5481
## 29 126             Taiwan      Asia  78.400   23174294 28718.2768
```

```
## 30 128           Thailand      Asia  70.616   65068149  7458.3963
## 31 138            Vietnam      Asia  74.249   85262356  2441.5764
## 32 139 West Bank and Gaza      Asia  73.422    4018332  3025.3498
## 33 140        Yemen, Rep.      Asia  62.698   22211743  2280.7699
```

The function 'func1' computes the mean, variance, and standard deviation of a vector of data.

```
func1(rnorm(10))
```

```
## $mean
## [1] 0.08547316
##
## $var
## [1] 0.8023862
##
## $sd
## [1] 0.8957601
```

The function 'func2' does the same thing as 'func1', with the addition that the supplied vector argument must be finite, numeric, and has a length greater than 0.

```
func2(rnorm(10))
```

```
## $mean
## [1] -0.3516461
##
## $var
## [1] 0.534199
##
## $sd
## [1] 0.7308892
```

The function 'func3' computed an estimate for the maximum likelihood estimator for data generated by the gamma distribution.

```
func3(rgamma(10, pi))
```

```
## [1] 3.567194
```

The function 'func4' computes the mean, variance, and standard deviation of a vector given a vector of weighted probabilities.

```
data(d)
head(d)
```

```
##            x          p
## 1 16.1904208 0.07413368
## 2  3.6366141 0.08652587
## 3 -2.5256698 0.01470391
## 4 10.3813887 0.01561383
## 5  0.9425428 0.08344035
## 6  4.3628222 0.05724653
```

```
func4(d)
```

```
## $mean
## [1] 4.940558
##
## $var
## [1] 16.12654
```

```
##
## $sd
## [1] 4.015786
```

The function 'func5' does the same thing as 'func4', with the addition that the supplied vectors must be finite, numeric, and has a length greater than 0.

```
d <- read.table(url("http://www.stat.umn.edu/geyer/3701/data/q1p4.txt"),header = TRUE)
func5(d)
```

```
## $mean
## [1] 4.940558
##
## $var
## [1] 16.12654
##
## $sd
## [1] 4.015786
```

The function 'func6' supplies an error message when the arguments supplied to a function is not viable.

```
func6(NA)
```

```
## [1] "not numeric"
## [1] "not finite"
## [1] "NA or NAN"
```

The function 'func7' computes the maximum likelihood estimate for a supplied data set generated by some distribution. In the following example, the data has been generated from the gamma distribution and the function implies that the maximum likelihood estimation should be done over the log likelihood function of the gamma distribution. The output returned in the maximum likelihood estimate of the parameter theta.

```
x1 <- ga_data
func1 = function(theta, x) dgamma(x, shape = theta, log = TRUE)
result7_gamma <- func7(x1,func1,c(0,3))
result7_gamma
```

```
## [1] 2.962396
```

The following functions were provided in Homework 2 and Quiz 2 of STAT 3701

The function 'xAx' and the binary operator '%xAx%' both computes the scalar value from $x^T A^{-1} x$, where x is an nx1 vector and A is a square nxn matrix.

```
load(url("http://www.stat.umn.edu/geyer/3701/data/q2p1.rda"))
a <- as.matrix(a)
x <- as.matrix(x)
xAx(a, x)
```

```
##      [,1]
## [1,]   13
```

```
a %xAx% x
```

```
##      [,1]
## [1,]   13
```

The function 'standardize' takes a matrix A and standardizes the columns of the matrix A. In this case, standardization of a column is computed by $\frac{x - \text{mean(x)}}{\text{sd(x)}}$, where $x \in A$, x is a column of the matrix A.

```
load(url("http://www.stat.umn.edu/geyer/3701/data/q2p3.rda"))
standardize(a)
```

```
##            [,1]       [,2]       [,3]        [,4]
## [1,]  0.7035431 -0.4380029  1.3181419 -0.37408291
## [2,] -1.0113432  0.8760058 -0.4524965  0.02200488
## [3,] -0.6156002  0.7117547 -0.5705390  1.21026823
## [4,]  1.4950291  1.0402568 -1.3968369 -0.24205365
## [5,] -0.8794289 -1.2592583  0.8459717 -1.56234625
## [6,]  0.3078001 -0.9307561  0.2557589  0.94620970
```

The function 'myapply' is a function that intends to replicate the results of the base R 'apply()' family functions. In this case, the function 'myapply' takes in args. myapply(MATRIX, MARGIN, FUNCTION) where MATRIX is a matrix, MARGIN is an integer 1 or 2 that corresponds to the row or column of the matrix respectively, and a FUNCTION function to act along the margins of the matrix.

```
fred <- matrix(1:6, ncol = 2)
myapply(fred, 1, mean)
```

```
## [1] 2.5 3.5 4.5
```

```
myapply(fred, 2, mean)
```

```
## [1] 2 5
```

```
myapply(fred, 1, max)
```

```
## [1] 4 5 6
```

```
myapply(fred, 2, max)
```

```
## [1] 3 6
```

```
myapply(fred, 1, function(x) quantile(x, 0.75))
```

```
##  75%  75%  75%
## 3.25 4.25 5.25
```

```
myapply(fred, 2, function(x) quantile(x, 0.75))
```

```
## 75% 75%
## 2.5 5.5
```

The wrapper 'p6wrapper()' returns the median values along a 3 dimensional data set.

```
load(url("http://www.stat.umn.edu/geyer/3701/data/q2p6.rda"))
pat
```

```
## , , color = red
##
##          computer
## transport Windows OSX Linux
##     car        99  35    17
##     truck      92  63    29
##
## , , color = white
##
##          computer
## transport Windows OSX Linux
##     car        34  18    97
```

```
##     truck      49  96    13
##
## , , color = blue
##
##           computer
## transport Windows OSX Linux
##     car          1  90    31
##     truck        5  70    40
##
## , , color = orange
##
##           computer
## transport Windows OSX Linux
##     car         98  91    93
##     truck       33  68    58
```

```
p6wrapper()
```

```
## [1] 41.5
## [1] 69
## [1] 35.5
```

ggplot2 Graphics

The following code chunk outputs three ways to display data presented in a TED talk given by Hans Rosling called "The best stats you've ever seen."

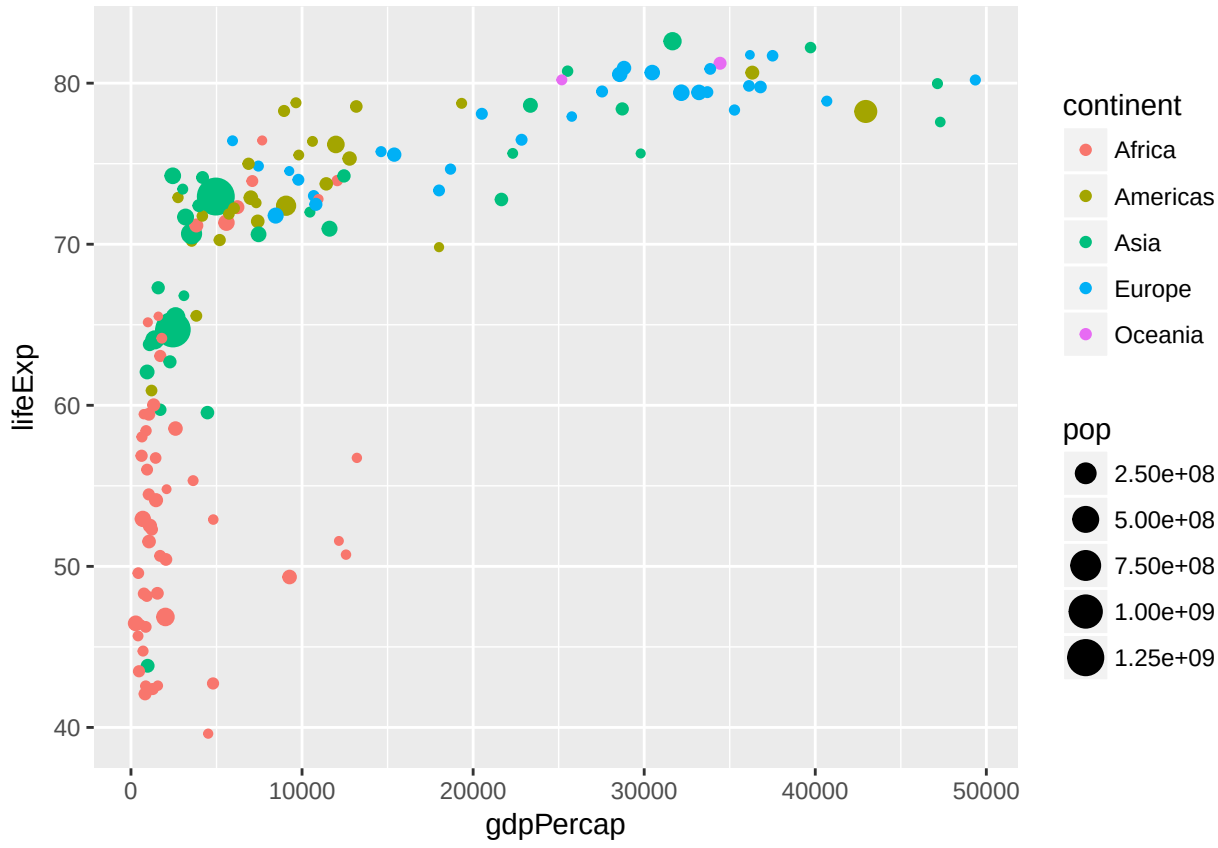
The variables used are 'gdpPercap' and 'lifeExp', which stand for Gross Domestic Product (in US dollars) and Life Expectancy in years, respectively.

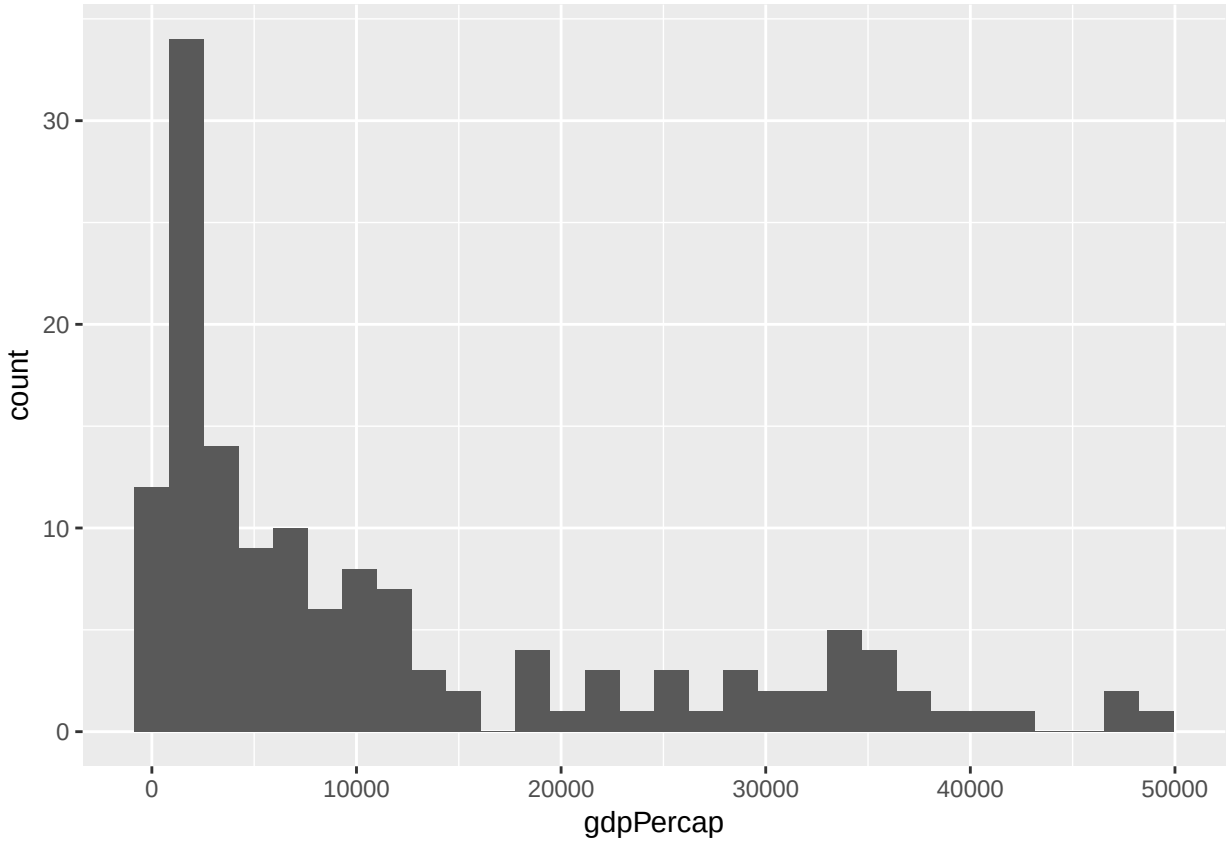
The plots were produced by using the package ggplot2.

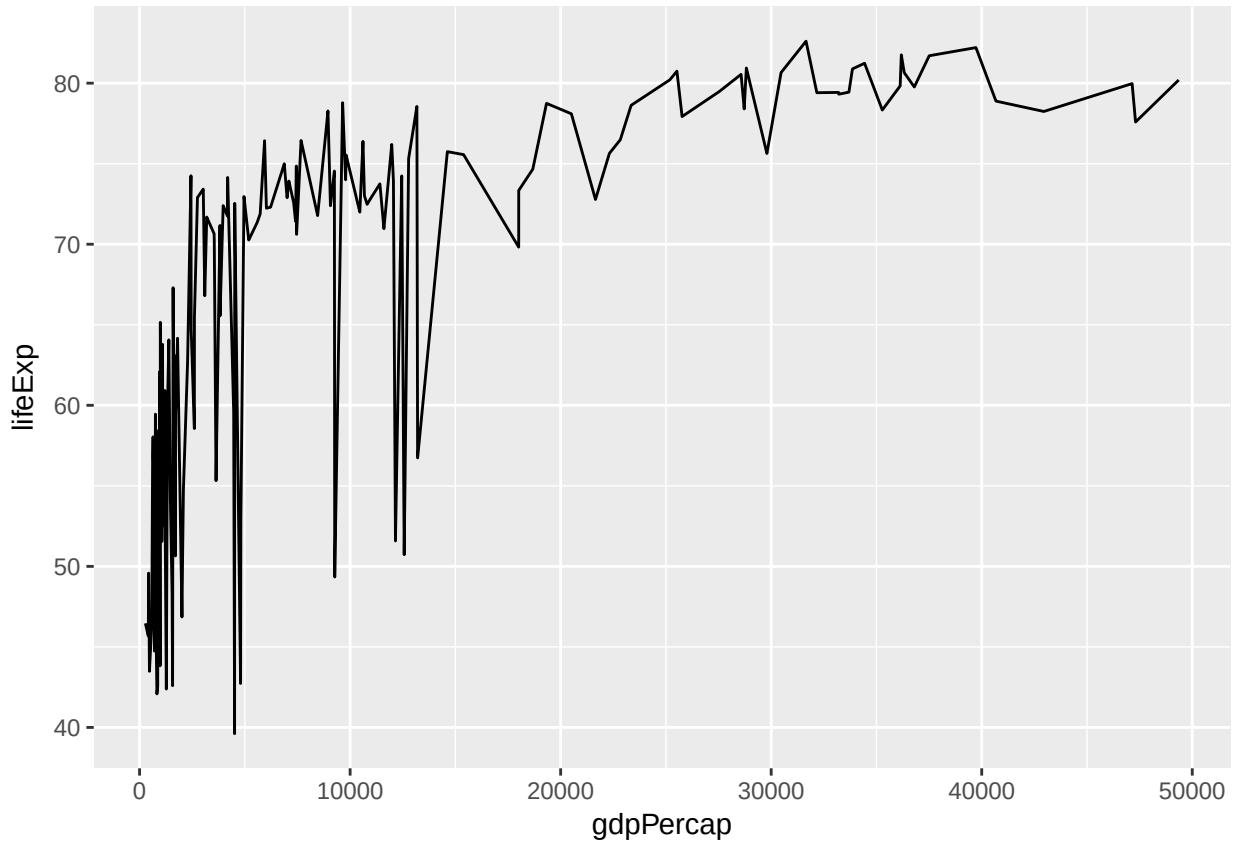
```
gp2007 <- read.csv('http://users.stat.umn.edu/~almquist/3811_examples/gapminder2007ex.csv')
ggplot2::ggplot(data=gp2007, ggplot2::aes(x=`gdpPercap`, y=`lifeExp`, size = `pop`, col = `continent`))
ggplot2::ggplot(data = gp2007, mapping = ggplot2::aes(x = gdpPercap)) + ggplot2::geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
ggplot2::ggplot(data = gp2007, ggplot2::aes(x = gdpPercap, y = lifeExp)) + ggplot2::geom_line()
```

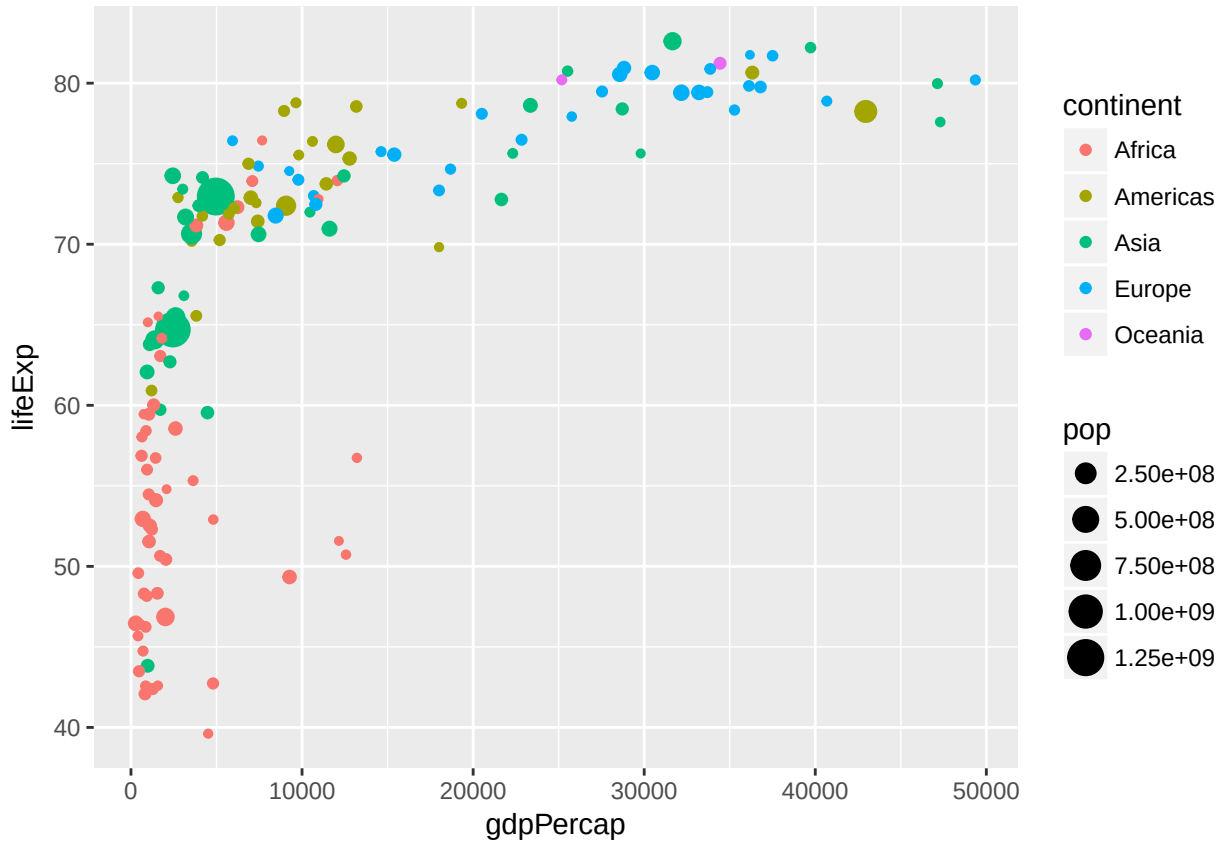

lifeExp
80
70
60
50
40
0
10000
20000
30000
40000
50000
gdpPercap
continent
Africa
Americas
Asia
Europe
Oceania
pop
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09


30
20
10
0
count
0
10000
20000
30000
40000
50000
gdpPercap



In Quiz 3, a requirement was to produce a wrapper that generated a plot of data. I will show the wrapper working below, which produces the plot above from the Rosling data.

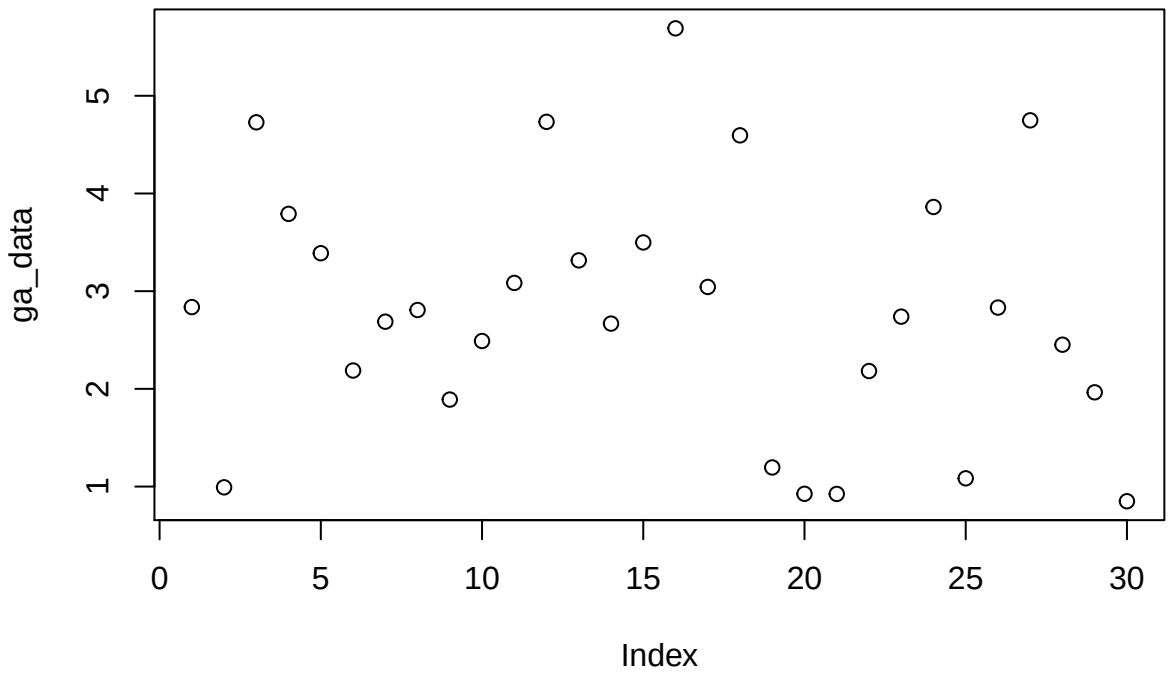
```
NguyenTools::plotstuff()
```

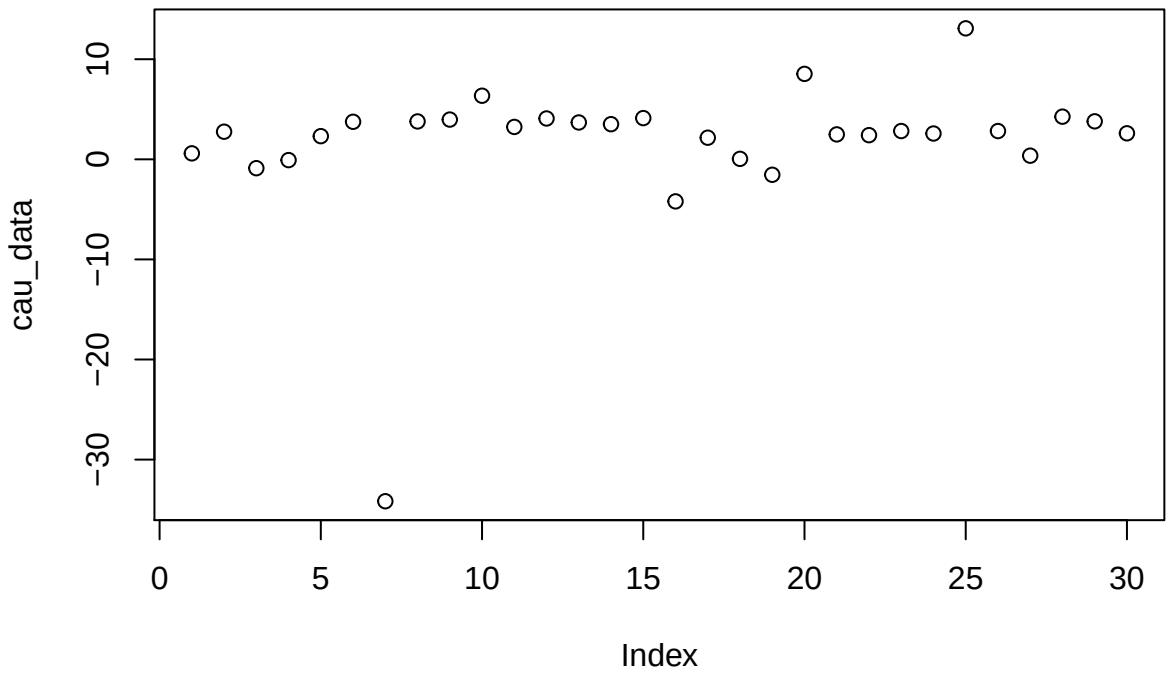


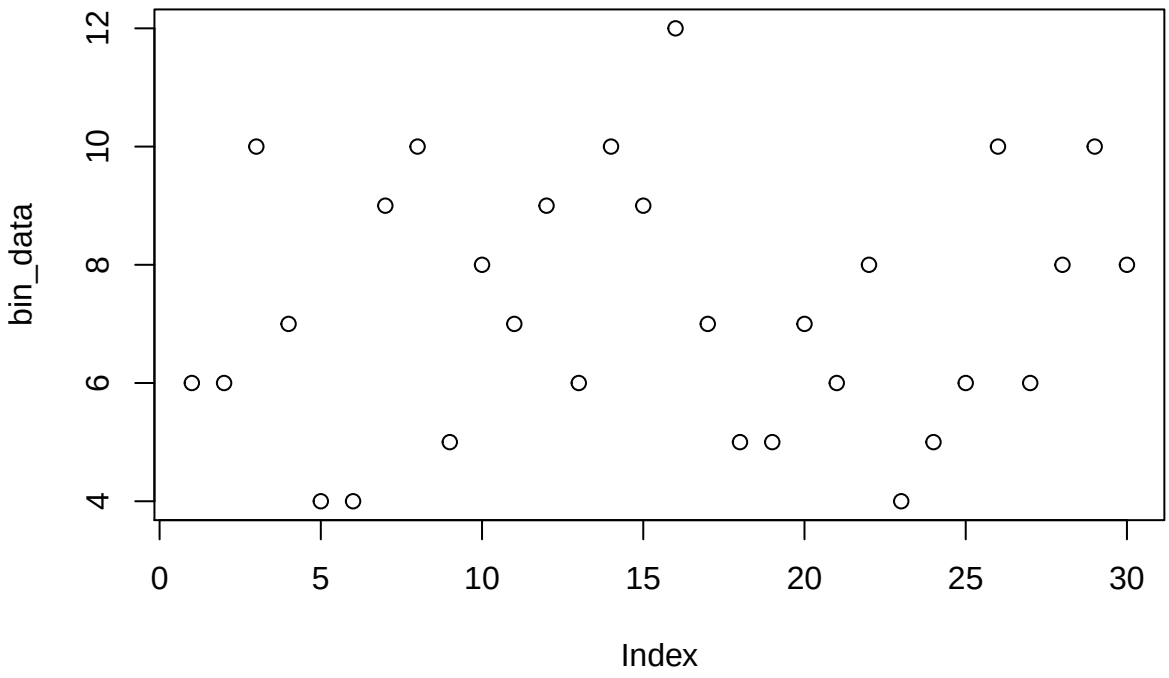
Plot of Data Sets included in the package

The data sets included in the package are binomial data, cauchy data, gamma data, alaskan flight data, early january weather data, life expectancy ~ gdp data, and weather data. The data was provided in STAT 3701 to use for various quizzes and homework assignments.

```
plot(ga_data)
plot(cau_data)
plot(bin_data)
```


ga_data
1
2
3
4
5
0
5
10
15
20
25
30
Index

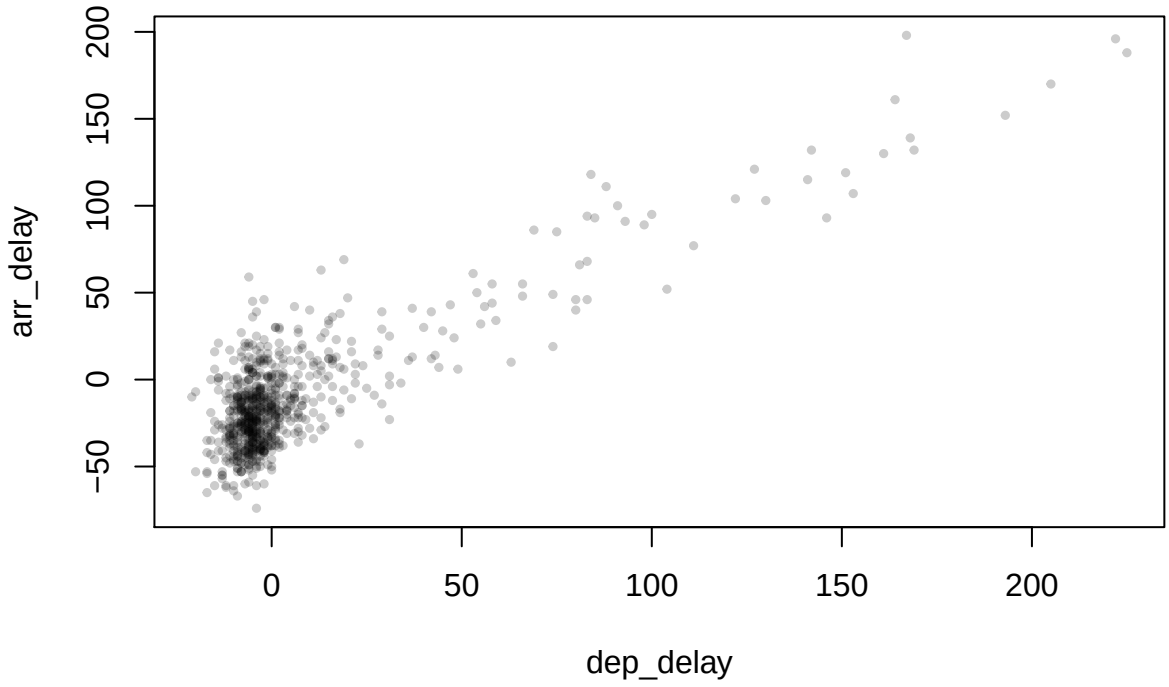

cau_data
10
0
−10
−20
−30
0
5
10
15
20
25
30
Index



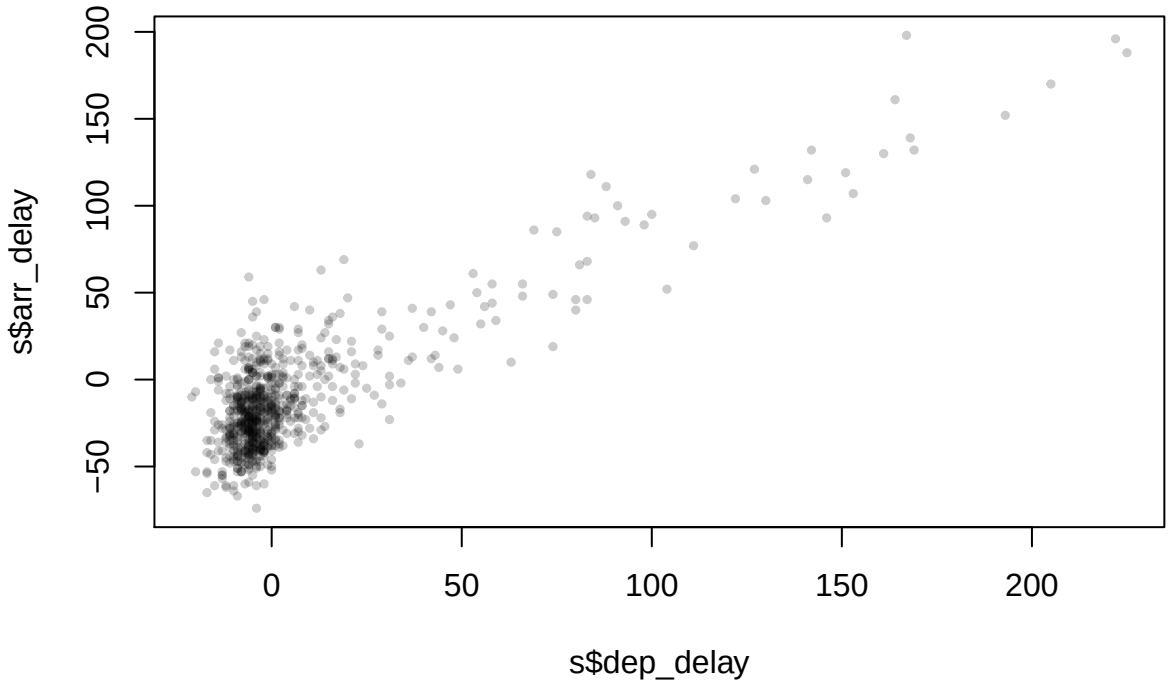
The plots for alaskan flight data, early january eather, and weather will be generated below. The plot for the dataset 'gp2007' has already been shown above, and will be omitted here.

The following chunk shows two ways to make the same plots for alaskan flight data.

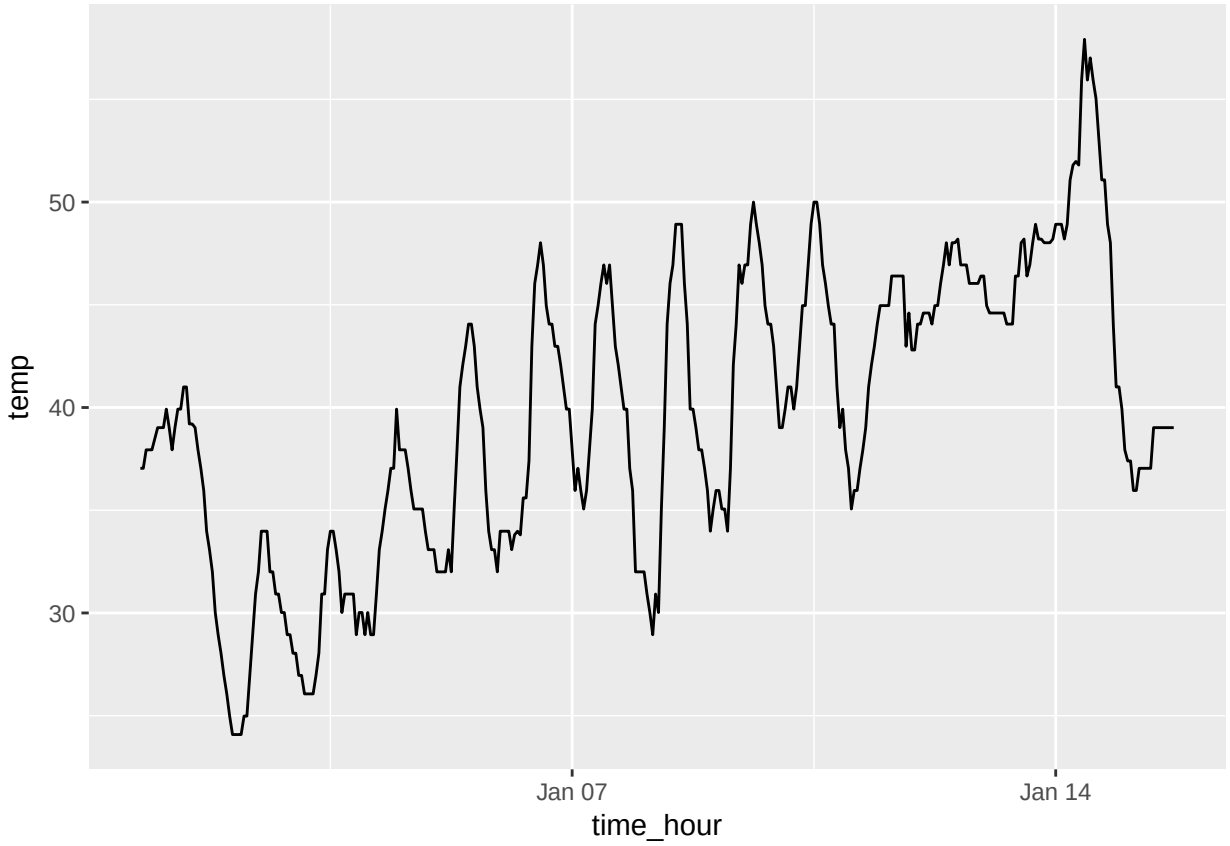
```
s <- all_alaska_flights
plot(arr_delay ~ dep_delay, data = s, pch = 19, cex = .5, col = rgb(0, 0, 0, .2) )
```




```
plot(s$dep_delay, s$arr_delay, pch = 19, cex = .5, col = rgb(0, 0, 0, .2) )
```

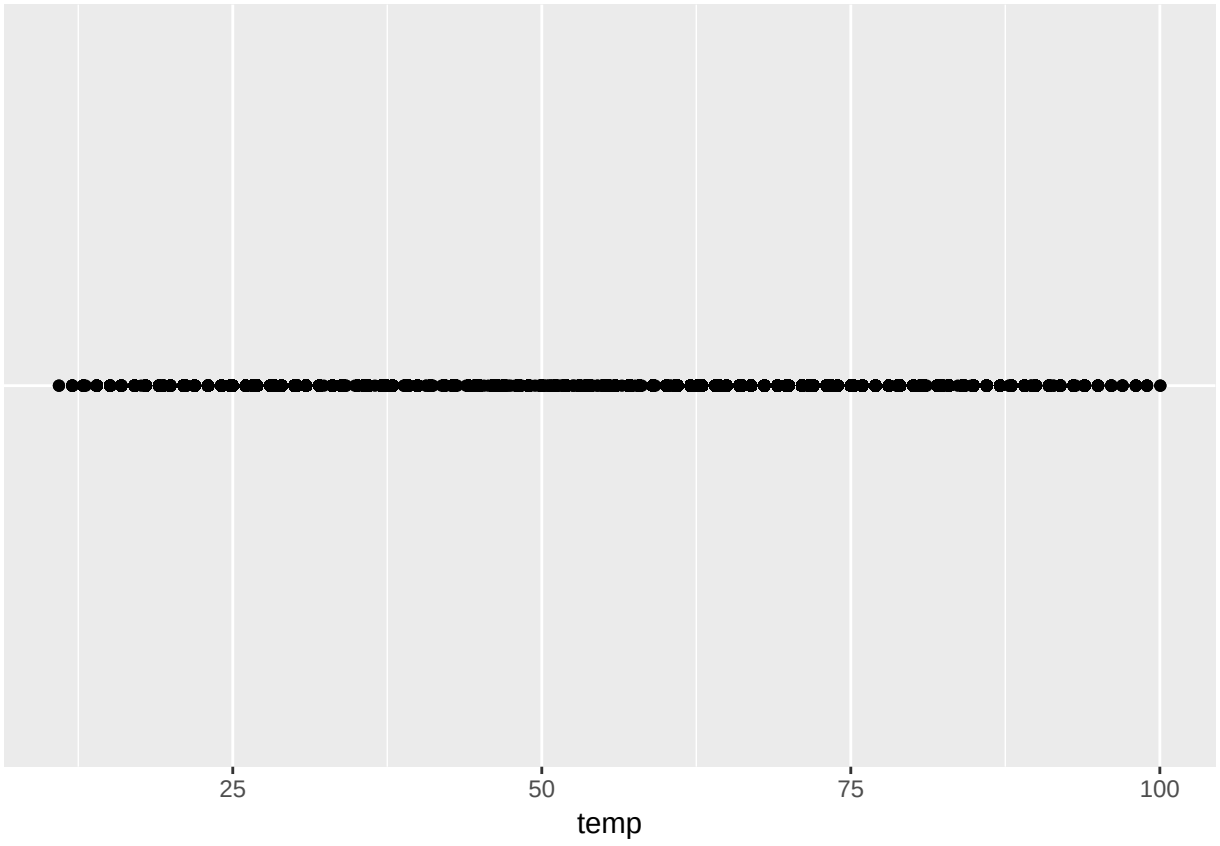



```
ggplot(data = early_january_weather, aes(x = time_hour, y = temp)) + geom_line()
```




```
ggplot(data = weather, mapping = aes(x = temp, y = factor("A"))) +
  geom_point() +
  theme(axis.ticks.y = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y = element_blank())
```

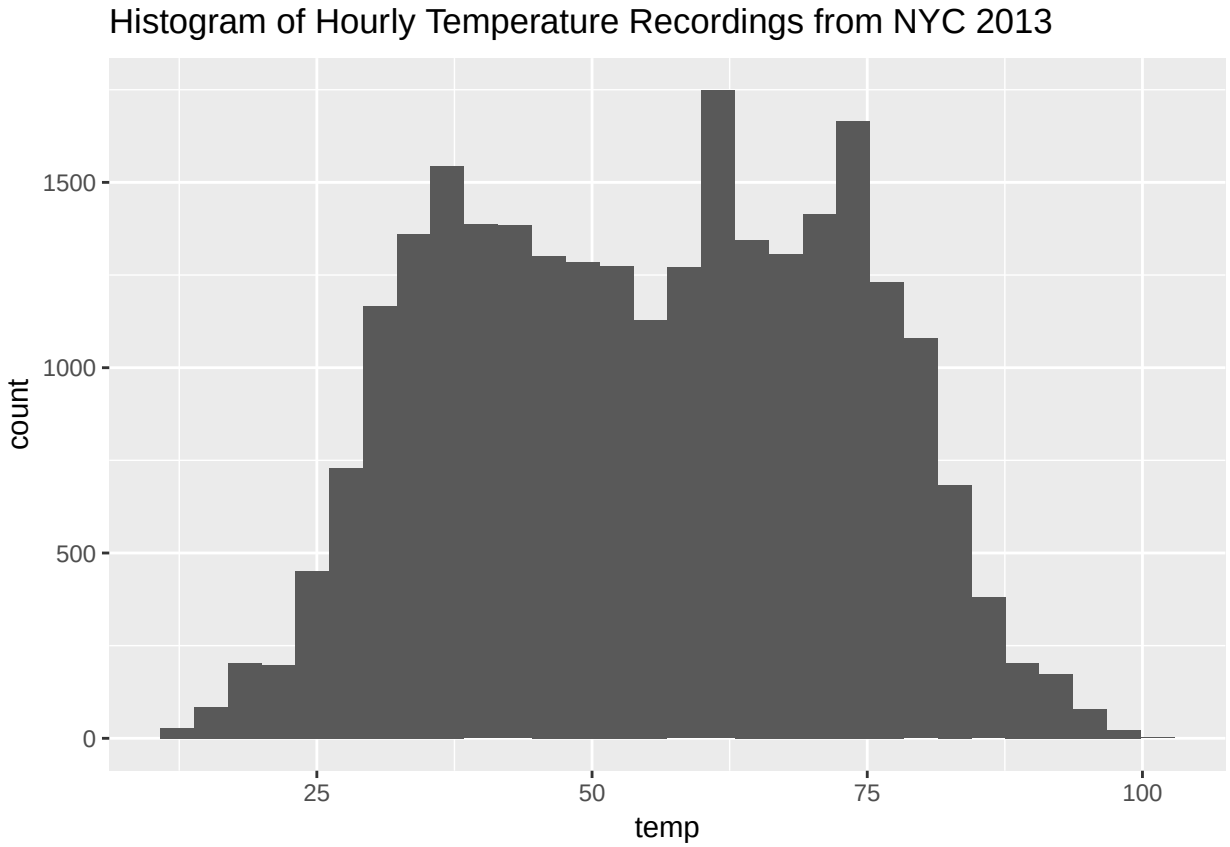
```
## Warning: Removed 1 rows containing missing values (geom_point).
```




```
hist_title <- "Histogram of Hourly Temperature Recordings from NYC 2013"
ggplot(data = weather, mapping = aes(x = temp)) + geom_histogram() + labs(title = hist_title)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing non-finite values (stat_bin).
```


Histogram of Hourly Temperature Recordings from NYC 2013
count
1500
1000
500
0
25
50
75
100
temp